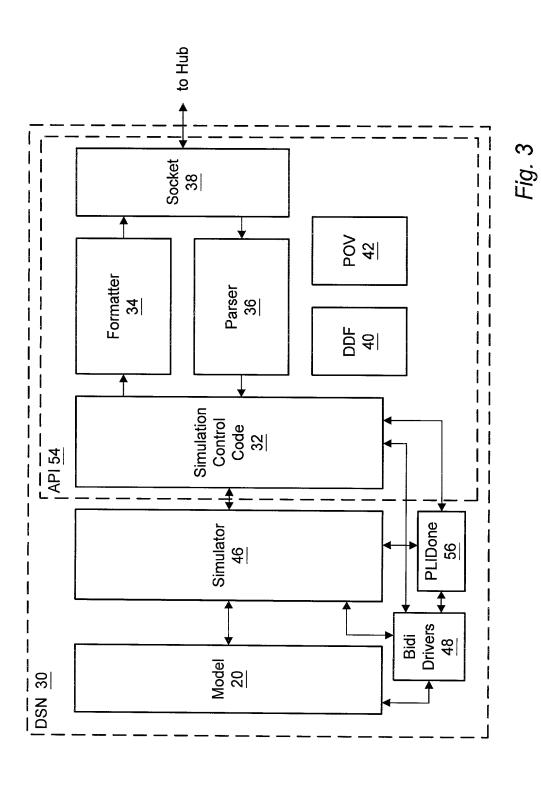
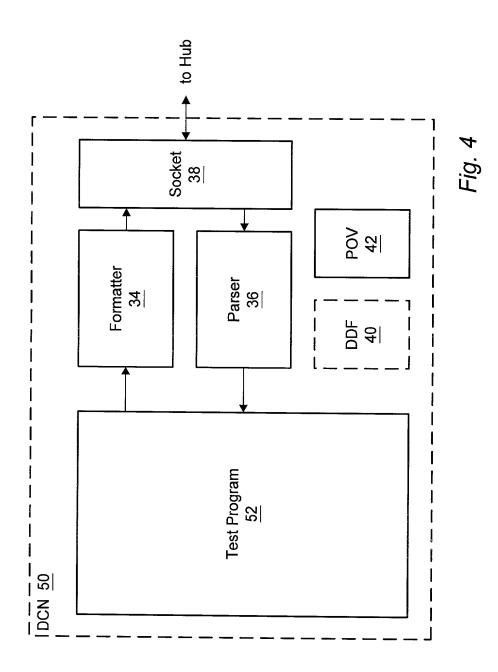
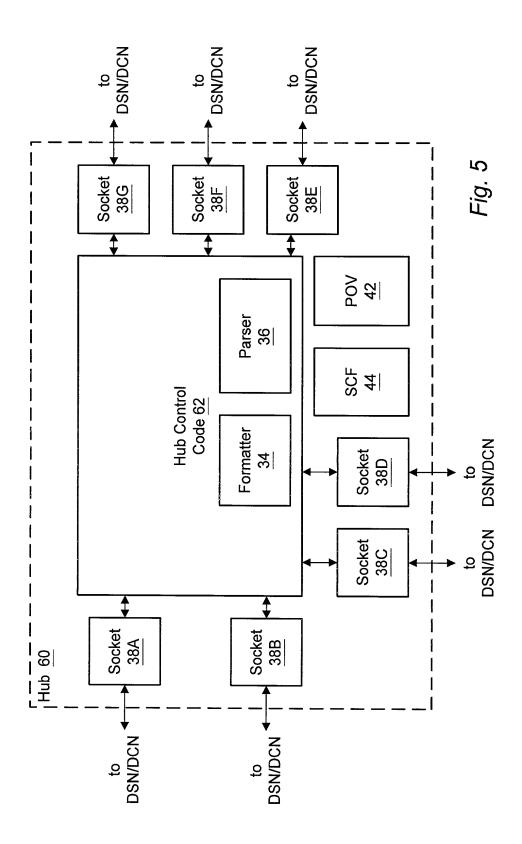
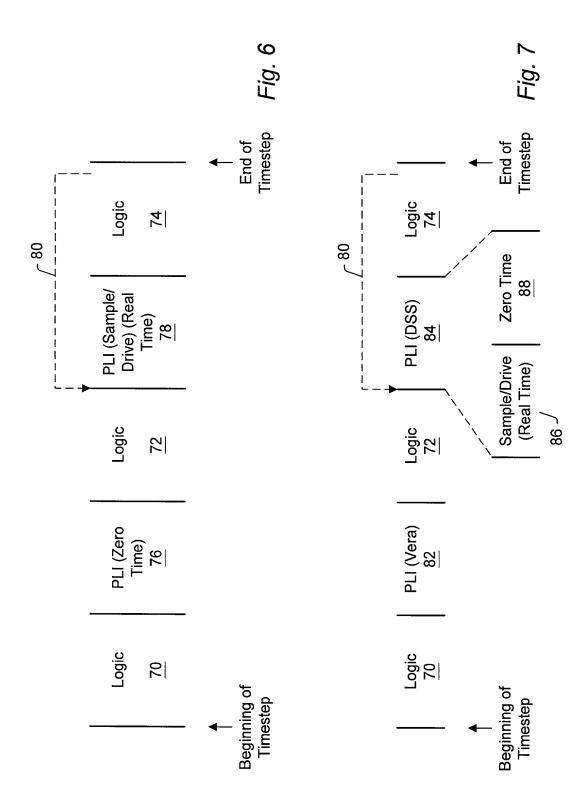


Fig. 2









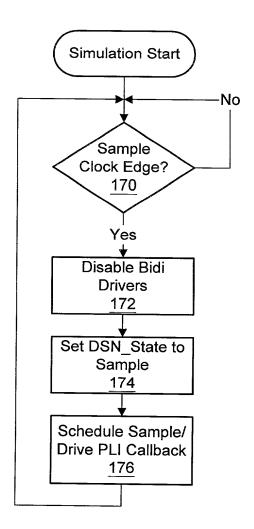
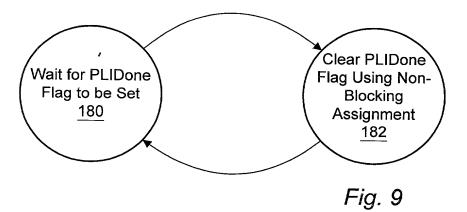
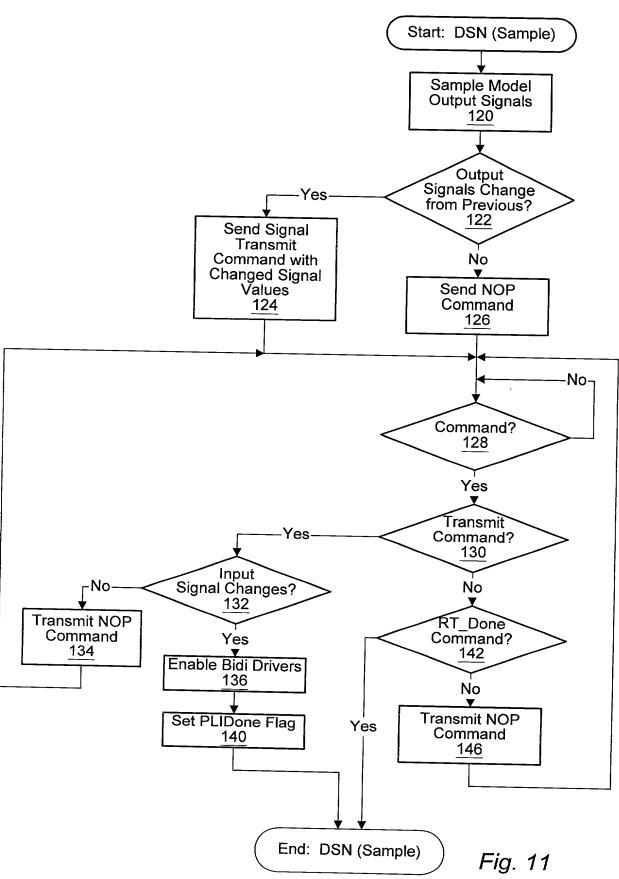


Fig. 8



Wait for PLIDone Flag to be Clear 184 DSN\_State No (Drive) = Sample? 186 Set DSN\_state to Sample 192 Yes Disable Bidi Set DSN\_state **Drivers** to Drive 194 188 Schedule Sample/ Drive PLI Callback 190 Fig. 10



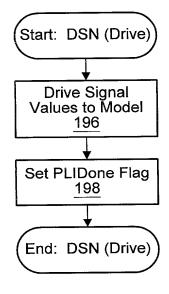


Fig. 12

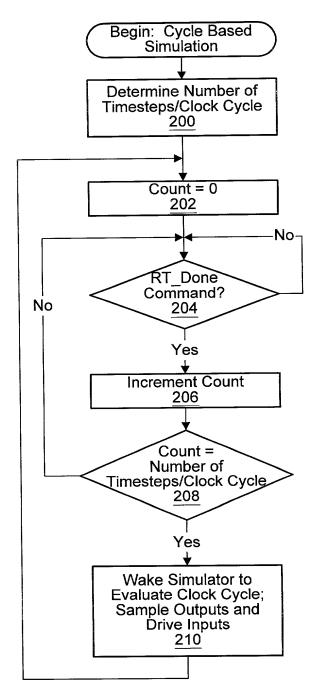


Fig. 13

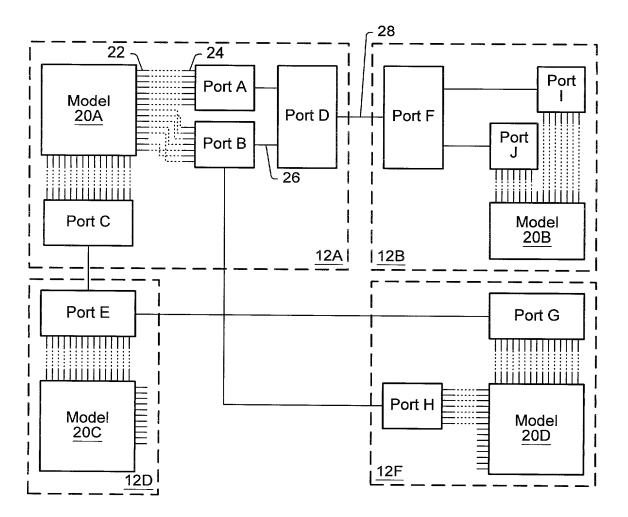


Fig. 14

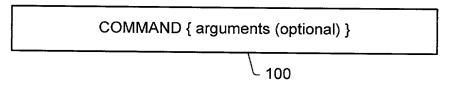


Fig. 15

Command	Arguments
POV	port definitions
SCF	model instances and routing
DDF	logical signal to physical signal mapping
TRANSMIT	source model instance, chip names, port names, signalnames, values, and strengths
NOP	source model instance
RT_DONE	source model instance
ZT_DONE	source model instance
ZT_FINISH	source model instance
FINISH	source model instance
USER	source model instance, user message
ERROR	source model instance, error message
HOTPLUG	source model instance, destination model instance
HOTPULL	source model instance, destination model instance
STOP	source model instance

Fig. 16

```
pov : POV '{' portdefs '}'
;
portdefs : | pordefs porttype '{' port_member more_members '}' ';'
porttype : PORTWORD BASENAME
;
port_member : SIGNALWORD NAME ';' | BASENAME NAME ';'
more_members : | more_members port_member
;
```

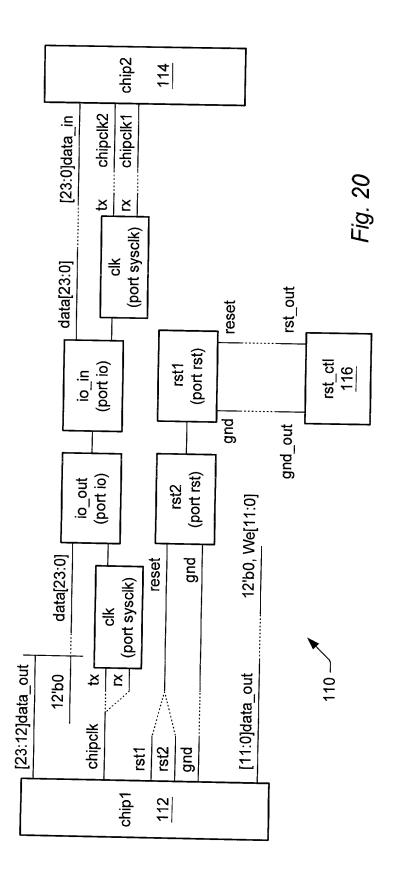
## Fig. 17

```
scf: SCF '{' scf_description '}'
scf_description: | model_instances routing
model_instances: one_instance more_instances
one_instance: BASENAME NAME ';'
more_instances: | more_instances one_instance
routing: | routing routing_exp
;
routing_exp: SCOPENAME1 '->' SCOPENAME1 ';'
```

Fig. 18

```
ddf: DDF '{' models '}'
models: | one model more models ';'
one model: BASENAME '{' log_phys '}' ';'
more models : | one model more models
log_phys: | logical physical
logical: LOGICAL '{' log_defs '}' ';'
log_defs: | BASENAME NAME ';'
physical: PHYSICAL '{' phys_defs '}' ';'
phys_defs: | one_to_one | many_to_one | one_to_many
one_to_one : SIGNALWORD signalname '{' signalparts '}' ';'
            one_to_one SIGNALWORD signalname '{' signalparts '}' ';'
signalname: BITWIDTH BASENAME | BASENAME
signalparts: one_signalpart more_signalparts
one signalpart : NAME '=' SCOPENAME2 ';'
more_signalparts: | more_signalparts one_signalpart
many_to_one : FORALL SIGNALWORD '(' sign_names ')' '{' logicalname '}' ';'
sig_names : signalname | sig_names ',' signalname
logicalname: SCOPENAME2 ';'
one_to_many: FOR SIGNALWORD '(' signalname ')' '{' signalparts '}' ';'
```

Fig. 19



```
POV {
                                              SCF {
       port io {
                                                      chip1 dsn1;
              signal data[23:0];
                                                      chip2 dsn2;
                                                      rst_ctl dsn3;
              sysclk clk;
              };
                                                      dsn1.io_out -> dsn2.io_in;
       port sysclk {
                                                      dsn3.rst1 -> dsn1.rst2;
              signal tx;
              signal rx;
                                                     }
              };
                                                          Fig. 22
       port rst {
              signal reset;
              signal gnd;
              };
       }
               Fig. 21
       DDF {
         chip1 {
          logical {
            io io_out;
            rst rst2;
          };
          physical {
            signal [23:0] data_out {
                   data_out[23:12] = io_out.data[23:12];
                   data_out[11:0] ={value: 12'b0, strength: We[11:0] };
             };
             signal gnd { gnd = rst2.gnd };
             for signal (chipclk) {
                    chipclk = io_out.clk.tx;
                    chipclk = io_out.clk.rx;
             };
             forall signals (rst1, rst2) { rst2.reset };
             signal default_logical { value: 12b'0 } = io_out.data[11:0]; };
          };
        };
       }
                                 Fig. 23
```

```
DDF {
   chip2 {
    logical {
      io io_in;
    };
    physical {
      signal [23:0] data_in {
            data_in[23:0] = io_in.data[23:0];
      };
      signal (chipclk1) {
             chipclk1 = io_in.clk.rx;
      };
      signal (chipclk2) {
             chipclk2 = io_in.clk.tx;
      };
   };
  };
}
                  Fig. 24
```

```
DDF {
    rst_ctl {
        logical {
            rst rst1;
        };
        physical {
            signal rst_out { rst_out = rst1.reset; };
            signal gnd_out { gnd_out = rst1.gnd; };
        };
    };
};
```

Fig. 25

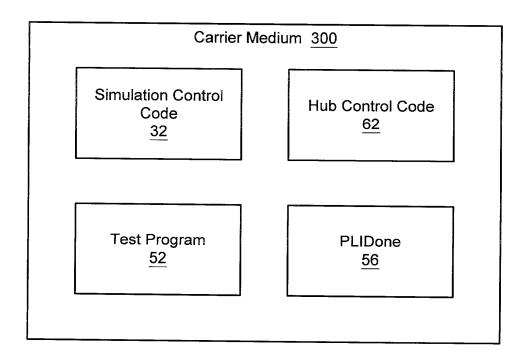


Fig. 26